# Appendix A

## Example Functional Architecture for Airborne Surveillance and Separation Assurance Processing (ASSAP)

# A Example Functional Architecture for Airborne Surveillance and Separation Assurance Processing (ASSAP)

## A.1 Introduction

This appendix outlines an example functional architecture for the ASSAP component of the Airborne Separation Assurance System (ASAS). The ASSAP subsystem performs the surveillance and application-specific processing functions of ASAS. The architecture presented in this appendix meets or exceeds the requirements listed in this document. Additionally, this architecture has been tested and validated with real-world and simulated data sets, including:

a. Operational 1090ES ADS-B, TIS-B, and TCAS data obtained from flight tests performed at the William J. Hughes [FAA] Technical Center in July 2007.

b. Simulated surveillance data based on RNAV approaches to Hartsfield-Jackson Atlanta International Airport in August 2006 (listed in sections TBD).

## A.2 Surveillance Processing

The ASSAP Surveillance Processor is required to establish tracks from ADS-B and TIS-B traffic reports, cross-reference traffic from different surveillance sources (ADS-B, TIS-B, TCAS), estimate track state (i.e. position, velocity) and track quality, and delete tracks when data age exceeds a maximum allowable coast time (dictated by applications). The Surveillance Processing function has been decomposed into the following subfunctions:

a. Track Generation and Maintenance

b. Inter-Source Correlation

c. Best Source Selection

d. Track Termination

e. Common Time Track Extrapolation

Figure A-1 contains a block diagram depicting the Surveillance Processing function.

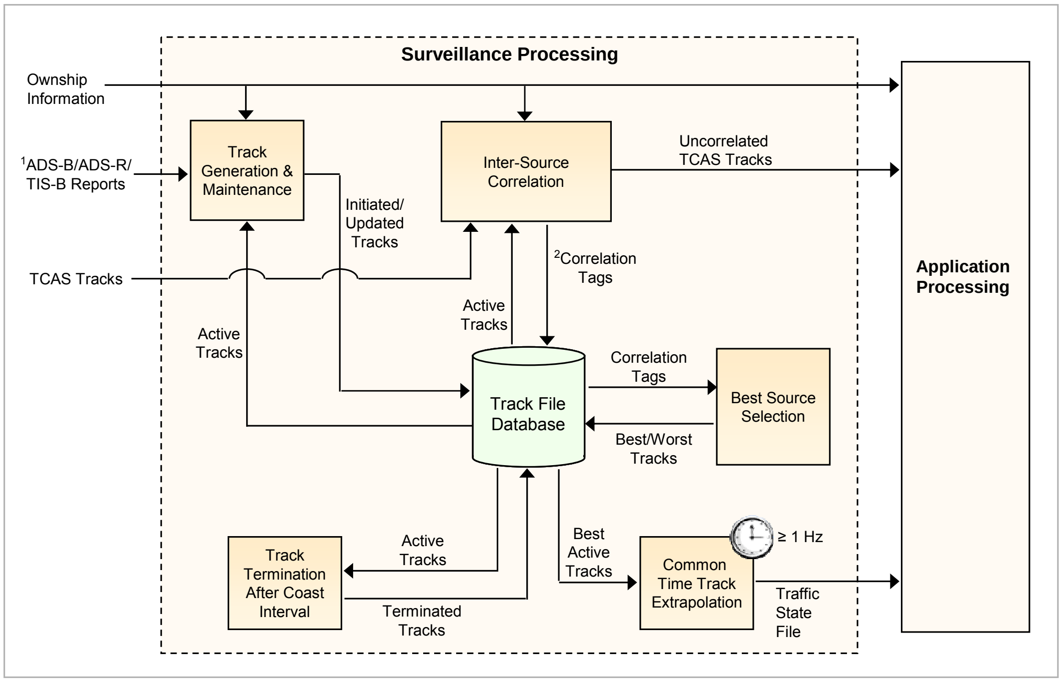


Figure A-1. Block diagram of the Surveillance Processing function.

*Notes:*

1) *Depending on the implementation (UAT or 1090ES), ASSAP will receive "full" reports (full state vector and state uncertainty) or "partial" reports (state vector and state uncertainty separately).*
2) *Correlation tags include TCAS correlation tags and track-to-track (i.e. ADS-B to TIS-B) correlation tags.*

## A.2.1 Track Generation and Maintenance

The Track Generation and Maintenance function is responsible for track initiation and subsequent updates. This function is an adaptation of a source-level tracker; ADS-B, ADS-R, and TIS-B tracks are established separately (and categorized by type). An effort was made to define this function in a manner applicable to both UAT and 1090ES implementations. However, UAT and 1090ES reports are inherently different, thus inputs to ASSAP are inherently different (depending on the implementation). Additionally, 1090ES characteristics and strict adherence to receiver standards (DO-260A) further [1]complicate inputs to ASSAP. As a result, the architecture in this appendix contains implementation-specific tracking functions.

*Note:*

1) *Although 1090ES receivers generate "state vector" reports (event-driven by the reception of a position or velocity message), position and velocity in any given report correspond to different times of applicability. Additionally, DO-260A standards allow a receiver to repeat a previous position if a new position message is not received, as long as the position time of applicability is encoded (and likewise for velocity). As observed during 1090ES flight tests at the FAA Technical Center, receivers will produce partially valid state vectors (i.e. valid velocity with invalid/repeated position).*

### A.2.1.1 Track Generation and Maintenance (1090ES Implementation)

Figure A-2 contains a block diagram depicting the Track Generation and Maintenance function for the 1090ES implementation of ASSAP.

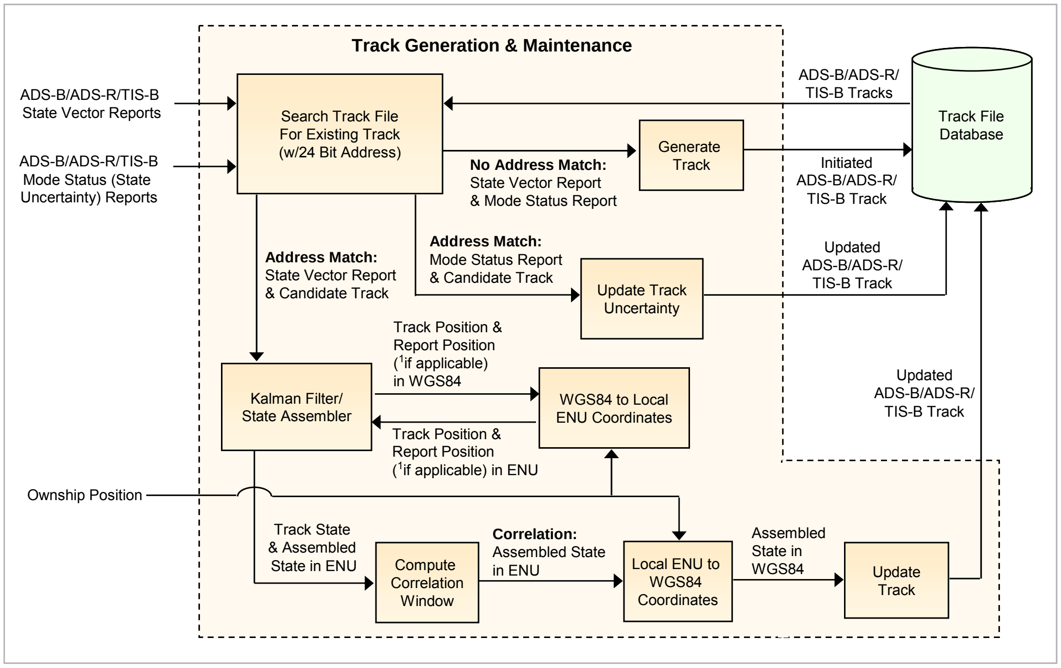


Figure A-2. Block diagram of the Track Generation and Maintenance function for 1090ES implementations of ASSAP.

*Note:*

1) *Reported position is applicable only if a position message was received, if a velocity message was received, velocity is used by the state assembler.*

#### A.2.1.1.1 Track Generation (1090ES Implementation)

Upon reception of a mode status report, ASSAP searches for a track of the same type (ADS-B, ADS-R, TIS-B) and with the same 24-bit address as the report. If no match is found, ASSAP generates a new track containing the following parameters:

a. System time (track time of applicability)
b. Track type (ADS-B, ADS-R, TIS-B)
c. 24-bit address
d. NACp
e. NACv
f. SIL
g. [1]Initialization status set to 0

*Note:*

1) *Initialization status is used to control processing of the first received state vector report (after track generation) and subsequent state vector reports.*

Upon reception of a state vector report, ASSAP searches for a track of the same type (ADS-B, ADS-R, TIS-B) and with the same 24-bit address as the report. If no match is found, the report is discarded. Note that a mode status report should arrive within milliseconds, generating a new track for the target and limiting the number of discarded state vector reports. If a match is found and track initialization status is set to 0, the track is "initialized" (prepared for subsequent updates) by adding/modifying following parameters:

a. System time (updated)
b. Initialization status set to 1
c. [1]Position (latitude, longitude, altitude)
d. [1]Velocity (East/West velocity, North/South velocity, altitude rate)
e. NIC
f. [2]State covariance matrices

*Notes:*

1) *Different times of applicability for position and velocity are disregarded during track initialization.*
2) *State covariance matrices are derived from NACp and NACv values stored in the track and pertain to x/y/z dimensions in the local East/North/Up (ENU) coordinate frame:*

*<u>Initial covariance matrix for the x-dimension</u>*

$$\begin{pmatrix} \sigma_x^2 & \sigma_{x\dot{x}} \\ \sigma_{x\dot{x}} & \sigma_{\dot{x}}^2 \end{pmatrix} = \begin{pmatrix} \sigma_{epu}^2 & \sigma_{epu}\sigma_{hva} \\ \sigma_{epu}\sigma_{hva} & \sigma_{hva}^2 \end{pmatrix}$$

*where*

$\sigma_{epu}$ *(standard deviation of estimated position uncertainty) is derived from NACp.*

$\sigma_{hva}$ *(standard deviation of horizontal velocity accuracy)is derived from NACv.*

*<u>Initial covariance matrix for the y-dimension</u>*

$$\begin{pmatrix} \sigma_y^2 & \sigma_{y\dot{y}} \\ \sigma_{y\dot{y}} & \sigma_{\dot{y}}^2 \end{pmatrix} = \begin{pmatrix} \sigma_{epu}^2 & \sigma_{epu}\sigma_{hva} \\ \sigma_{epu}\sigma_{hva} & \sigma_{hva}^2 \end{pmatrix}$$

*<u>Initial covariance matrix for the z-dimension</u>*

$$\begin{pmatrix} \sigma_z^2 & \sigma_{z\dot{z}} \\ \sigma_{z\dot{z}} & \sigma_{\dot{z}}^2 \end{pmatrix} = \begin{pmatrix} \sigma_{vepu}^2 & \sigma_{vepu}\sigma_{vva} \\ \sigma_{vepu}\sigma_{vva} & \sigma_{vva}^2 \end{pmatrix}$$

*where*

$\sigma_{vepu}$ *(standard deviation of vertical estimated position uncertainty) is derived from NACp if geometric altitude is used and NACp ≥ 9, or an assumed value otherwise.*

$\sigma_{vva}$ *(standard deviation of vertical velocity accuracy)is derived from NACv if geometric altitude is used, or an assumed value otherwise.*

### A.2.1.1.2 Track Maintenance (1090ES Implementation)

Upon reception of a mode status update, ASSAP searches for a track of the same type (ADS-B, ADS-R, TIS-B) and with the same 24-bit address as the report. If a match is found, NACp, NACv, and SIL values in the track are updated with the values in the report.

Upon reception of a state vector update, ASSAP searches for a track of the same type (ADS-B, ADS-R, TIS-B) and with the same 24-bit address as the report. If a match is found and track initialization status is set to 1, the reported state component (position or velocity) with the lowest data age (determined with respective times of applicability) is sent to the state assembler function along with candidate track information.

#### A.2.1.1.2.1 State Assembler

The state assembly function uses three independent, two state Kalman filters. The filter state consists of position and velocity. The state is tracked in each of three orthogonal Cartesian dimensions (local ENU). The filter takes as input the measured position or velocity, the candidate track state, and the candidate track state covariance. The filter also requires as input the time of the measurement. The filter produces an "assembled" state vector and covariance. Regardless of update type (position or velocity), the state assembler temporarily (i.e. in memory) converts track position from WGS84 to local ENU coordinates:

*Track and ownship positions are first converted from WGS84 to ECEF coordinates:*

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \left(\frac{a}{\chi} + h\right)\cos\phi\cos\lambda \\ \left(\frac{a}{\chi} + h\right)\cos\phi\sin\lambda \\ \left(\frac{a(1-e^2)}{\chi} + h\right)\sin\phi \end{bmatrix} \qquad (1)$$

*where*

$a = semi\text{-}major\ axis\ =\ 6378137.0\ \ meters$

$e^2 = first\ eccentricity\ squared\ =\ 6.69437999014\ x\ 10^{-3}$

$\phi = latitude$

$\lambda = longitude$

$h = altitude$

$\chi = \sqrt{1 - e^2 \sin^2\phi}$

*Track ENU position is derived from track and ownship ECEF positions:*

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} -\sin\lambda & \cos\lambda & 0 \\ -\sin\phi'\cos\lambda & -\sin\phi'\sin\lambda & \cos\phi' \\ \cos\phi'\cos\lambda & \cos\phi'\sin\lambda & \sin\phi' \end{bmatrix} \begin{bmatrix} X_t - X_o \\ Y_t - Y_o \\ Z_t - Z_o \end{bmatrix} \qquad (2)$$

*where*

$$\phi' = \tan^{-1}\left(\frac{Z_o}{\sqrt{X_o^2 + Y_o^2}}\right) = \text{ geocentric latitude of ownship}$$
$(X_o, Y_o, Z_o) =$ *ECEF position of ownship*
$(X_t, Y_t, Z_t) =$ *ECEF position of target*

Track state (in local ENU) is then extrapolated to the time of applicability of the state vector report:

$$\begin{aligned}
\hat{x} &= x + \dot{x}(dt) \\
\hat{y} &= y + \dot{y}(dt) \\
\hat{z} &= z + \dot{z}(dt) \\
\hat{\dot{x}} &= \dot{x} \\
\hat{\dot{y}} &= \dot{y} \\
\hat{\dot{z}} &= \dot{z}
\end{aligned} \tag{3}$$

*where*

$(x, y, z)$ *is track position in local ENU coordinates*
$(\dot{x}, \dot{y}, \dot{z})$ *is track velocity (East/West, North/South, Altitude velocity components)*
$(\hat{x}, \hat{y}, \hat{z})$ *is predicted track position in local ENU coordinates*
$(\hat{\dot{x}}, \hat{\dot{y}}, \hat{\dot{z}})$ *is predicted track velocity*
*dt is the time difference between the current report and the last track update.*

The next step in the filtering process is the extrapolation of track covariance matrices (note that only formulas for the x/east dimension are shown, as they are identical for all dimensions):

$$\begin{aligned}
\sigma_{\hat{x}}^2 &= \sigma_x^2 + (dt)^2 \sigma_{\dot{x}}^2 + 2dt\sigma_{x\dot{x}} + \frac{Q(dt)^4}{4} \\
\sigma_{\hat{\dot{x}}}^2 &= \sigma_{\dot{x}}^2 + (dt)^2 Q \\
\sigma_{\hat{x}\hat{\dot{x}}} &= \sigma_{x\dot{x}} + (dt)\sigma_{\dot{x}}^2 + \frac{(dt)^3 Q}{2}
\end{aligned} \tag{4}$$

*where*

*Q is the process (plant) noise variance. Typical values are on the order of* $0.0025g^2$ *(with* $g = 9.8\ m/s^2$*). However, extensive simulation testing has shown that* $Q = 0.0625g^2$ *is optimal for maneuvering aircraft.*

#### A.2.1.1.2.1.1 State Assembly With Position Updates

Equations (1) and (2) are used to convert a position update from WGS84 to local ENU coordinates. Measurement position variances are derived from the NACp value stored in the candidate track. It is important to note that this is the technique used to link state vector and mode status reports. It is likely that a mode status report updated NACp and NACv in the track milliseconds before the arrival of the state vector update. First, the residual (or innovation) variances are calculated with extrapolated track position variances and estimated measurement position variances (based on most current NACp):

$$\begin{aligned} \sigma_{v_x}^2 &= \sigma_{\hat{x}}^2 + \sigma_{epu}^2 \\ \sigma_{v_y}^2 &= \sigma_{\hat{y}}^2 + \sigma_{epu}^2 \\ \sigma_{v_z}^2 &= \sigma_{\hat{z}}^2 + \sigma_{vepu}^2 \end{aligned} \tag{5}$$

*where*

$\left(\sigma_{\hat{x}}^2, \sigma_{\hat{y}}^2, \sigma_{\hat{z}}^2\right)$ *are extrapolated track position variances*

$\left(\sigma_{epu}^2, \sigma_{vepu}^2\right)$ *are estimated measurement position variances derived from NACp*

Gain vectors are then calculated with extrapolated track and residual variances:

$$\begin{aligned} w_{0_x} &= \frac{\sigma_{\hat{x}}^2}{\sigma_{v_x}^2}, \; w_{1_x} = \frac{\sigma_{\hat{x}\hat{\dot{x}}}}{\sigma_{v_x}^2} \\ w_{0_y} &= \frac{\sigma_{\hat{y}}^2}{\sigma_{v_y}^2}, \; w_{1_y} = \frac{\sigma_{\hat{y}\hat{\dot{y}}}}{\sigma_{v_y}^2} \\ w_{0_z} &= \frac{\sigma_{\hat{z}}^2}{\sigma_{v_z}^2}, \; w_{1_z} = \frac{\sigma_{\hat{z}\hat{\dot{z}}}}{\sigma_{v_z}^2} \end{aligned} \tag{6}$$

The assembled state is generated as follows:

$$\begin{aligned} x_a &= \hat{x} + w_{0x}(x_m - \hat{x}) \\ y_a &= \hat{y} + w_{0_y}(y_m - \hat{y}) \\ z_a &= \hat{z} + w_{0z}(z_m - \hat{z}) \\ \dot{x}_a &= \hat{\dot{x}} + w_{1x}(x_m - \hat{x}) \\ \dot{y}_a &= \hat{\dot{y}} + w_{1y}(y_m - \hat{y}) \\ \dot{z}_a &= \hat{\dot{z}} + w_{1z}(z_m - \hat{z}) \end{aligned} \tag{7}$$

*where*

$\left(x_m, y_m, z_m\right)$ *are measurement positions in local ENU*

The assembled state covariance matrices are calculated as follows:

$$
\begin{aligned}
\sigma_{x_a}^2 &= (1-w_{0_x})\sigma_{\hat{x}}^2 \\
\sigma_{x\dot{x}a} &= (1-w_{0_x})\sigma_{\hat{x}\hat{\dot{x}}} \\
\sigma_{\dot{x}_a}^2 &= \sigma_{\hat{\dot{x}}}^2 - w_{1_x}\sigma_{\hat{x}\hat{\dot{x}}} \\
\\
\sigma_{y_a}^2 &= (1-w_{0_y})\sigma_{\hat{y}}^2 \\
\sigma_{y\dot{y}a} &= (1-w_{0_y})\sigma_{\hat{y}\hat{\dot{y}}} \\
\sigma_{\dot{y}_a}^2 &= \sigma_{\hat{\dot{y}}}^2 - w_{1_y}\sigma_{\hat{y}\hat{\dot{y}}} \\
\\
\sigma_{z_a}^2 &= (1-w_{0_z})\sigma_{\hat{z}}^2 \\
\sigma_{z\dot{z}a} &= (1-w_{0_z})\sigma_{\hat{z}\hat{\dot{z}}} \\
\sigma_{\dot{z}_a}^2 &= \sigma_{\hat{\dot{z}}}^2 - w_{1_z}\sigma_{\hat{z}\hat{\dot{z}}}
\end{aligned}
\tag{8}
$$

The assembled state and track state (both in local ENU coordinates) are then sent to the correlation window function.

### A.2.1.1.2.1.2 State Assembly With Velocity Updates

In the case of a velocity update, the residual (or innovation) variances are calculated with extrapolated track velocity variances and estimated measurement velocity variances (based on most current NACv):

$$
\begin{aligned}
\sigma_{v_x}^2 &= \sigma_{\hat{\dot{x}}}^2 + \sigma_{hva}^2 \\
\sigma_{v_y}^2 &= \sigma_{\hat{\dot{y}}}^2 + \sigma_{hva}^2 \\
\sigma_{v_z}^2 &= \sigma_{\hat{\dot{z}}}^2 + \sigma_{vva}^2
\end{aligned}
\tag{9}
$$

*where*

$\left(\sigma_{\hat{\dot{x}}}^2, \sigma_{\hat{\dot{y}}}^2, \sigma_{\hat{\dot{z}}}^2\right)$ *are extrapolated track velocity variances*

$\left(\sigma_{hva}^2, \sigma_{vva}^2\right)$ *are estimated measurement velocity variances derived from NACv*

Gain vectors are then calculated with extrapolated track and residual variances:

$$
\begin{aligned}
w_{0_x} &= \frac{\sigma_{\hat{x}\hat{\dot{x}}}}{\sigma_{v_x}^2},\ w_{1_x} = \frac{\sigma_{\hat{\dot{x}}}^2}{\sigma_{v_x}^2} \\
w_{0_y} &= \frac{\sigma_{\hat{y}\hat{\dot{y}}}}{\sigma_{v_y}^2},\ w_{1_y} = \frac{\sigma_{\hat{\dot{y}}}^2}{\sigma_{v_y}^2} \\
w_{0_z} &= \frac{\sigma_{\hat{z}\hat{\dot{z}}}}{\sigma_{v_z}^2},\ w_{1_z} = \frac{\sigma_{\hat{\dot{z}}}^2}{\sigma_{v_z}^2}
\end{aligned}
\tag{10}
$$

The assembled state is generated as follows:

$$
\begin{aligned}
x_a &= \hat{x} + w_{0_x}(\dot{x}_m - \hat{\dot{x}}) \\
y_a &= \hat{y} + w_{0_y}(\dot{y}_m - \hat{\dot{y}}) \\
z_a &= \hat{z} + w_{0_z}(\dot{z}_m - \hat{\dot{z}}) \\
\dot{x}_a &= \hat{\dot{x}} + w_{1_x}(\dot{x}_m - \hat{\dot{x}}) \\
\dot{y}_a &= \hat{\dot{y}} + w_{1_y}(\dot{y}_m - \hat{\dot{y}}) \\
\dot{z}_a &= \hat{\dot{z}} + w_{1_z}(\dot{z}_m - \hat{\dot{z}})
\end{aligned}
\tag{11}
$$

*where*

$(\dot{x}_m, \dot{y}_m, \dot{z}_m)$ *are measurement East/West, North/South, and altitude velocities (respectively)*

Next, the assembled state covariance matrices are calculated as follows:

$$
\begin{aligned}
\sigma_{x_a}^2 &= \sigma_{\hat{x}}^2 - w_{0_x}\sigma_{\hat{x}\hat{\dot{x}}} \\
\sigma_{x\dot{x}_a} &= (1 - w_{1_x})\sigma_{\hat{x}\hat{\dot{x}}} \\
\sigma_{\dot{x}_a}^2 &= (1 - w_{1_x})\sigma_{\hat{\dot{x}}}^2 \\[1em]
\sigma_{y_a}^2 &= \sigma_{\hat{y}}^2 - w_{0_y}\sigma_{\hat{y}\hat{\dot{y}}} \\
\sigma_{y\dot{y}_a} &= (1 - w_{1_y})\sigma_{\hat{y}\hat{\dot{y}}} \\
\sigma_{\dot{y}_a}^2 &= (1 - w_{1_y})\sigma_{\hat{\dot{y}}}^2 \\[1em]
\sigma_{z_a}^2 &= \sigma_{\hat{z}}^2 - w_{0_z}\sigma_{\hat{z}\hat{\dot{z}}} \\
\sigma_{z\dot{z}_a} &= (1 - w_{1_z})\sigma_{\hat{z}\hat{\dot{z}}} \\
\sigma_{\dot{z}_a}^2 &= (1 - w_{1_z})\sigma_{\hat{\dot{z}}}^2
\end{aligned}
\tag{12}
$$

The assembled state and track state (both in local ENU coordinates) are then sent to the correlation window function.

#### A.2.1.1.2.2 Correlation Window

The correlation window is used to perform a final check that ensures the received state vector report truly corresponds to the candidate track.  The correlation window is an estimated maximum distance between two reported positions, based on the concepts shown in Figure A-3.

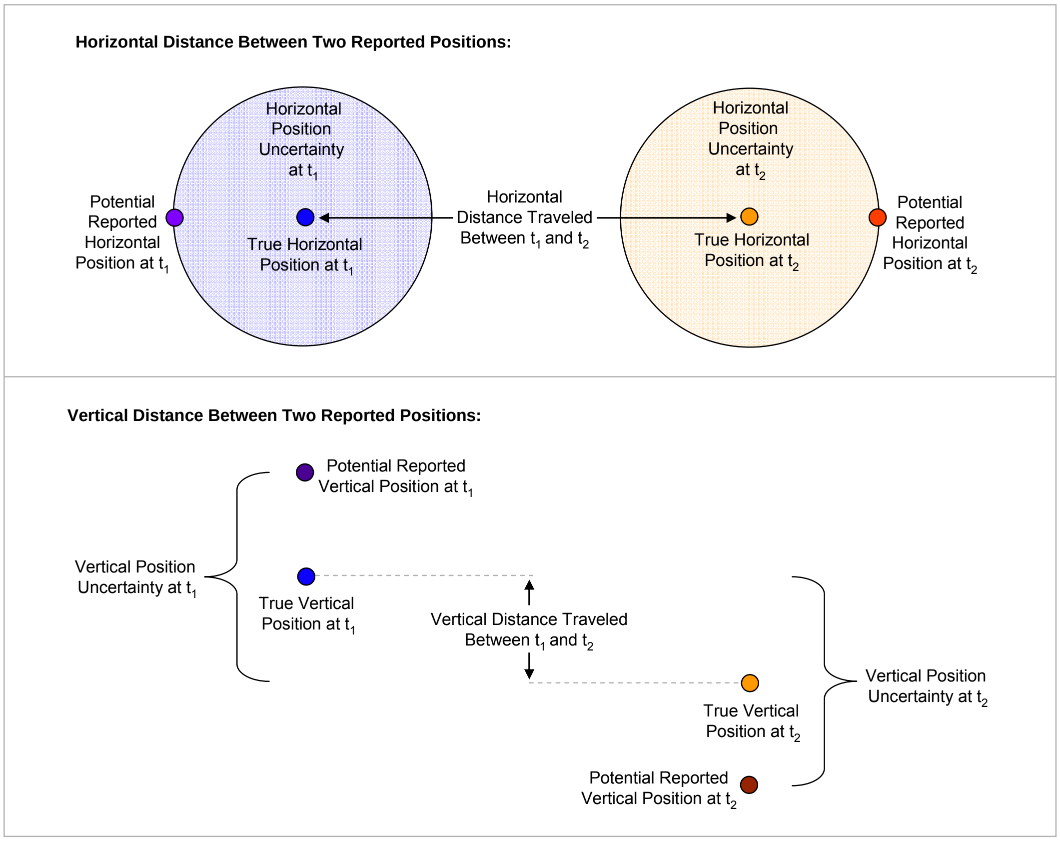


Figure A-3. Estimated Maximum Horizontal and Vertical Distances Between Two Reported Positions.

The correlation window is calculated as follows:

$$
\begin{aligned}
r_h &= \sqrt{\dot{x}_a^2 + \dot{y}_a^2}\,(dt) + 2(3\sigma'_{epu}) \\
r_v &= \left|\dot{z}_a\right| dt + 2(3\sigma'_{vepu}) \\
r_s &= \sqrt{r_h^2 + r_v^2}
\end{aligned}
\qquad (13)
$$

*where*

$\sigma'_{epu}$ , $\sigma'_{vepu}$ *are derived from degraded track NACp (NACp -1)*

$r_h$ *is estimated maximum horizontal distance between the assembled and track positions*

$r_v$ *is estimated maximum vertical distance between the assembled and track positions*

$r_s$ *is the correlation window, or estimated maximum slant range between the assembled and track positions*

*Note that these estimates assume linear dynamics (no acceleration)*

Next, slant range between assembled and track positions is calculated:

$$
\begin{aligned}
dx &= x_a - x \\
dy &= y_a - y \\
dz &= z_a - z \\
r &= \sqrt{dx^2 + dy^2 + dz^2}
\end{aligned}
\tag{14}
$$

*where*

$(dx, dy, dz)$ *is the difference between assembled/filtered and track positions.*

If the slant range between assembled and track positions is less than or equal to the maximum estimated slant range ($r \le r_s$), spatial correlation occurs.

### A.2.1.1.2.3 Track Updates

Upon successful correlation of assembled and track states, the assembled state is converted from local ENU to WGS84 coordinates:

*Ownship position is first converted from WGS84 to ECEF coordinates using equation (1). The assembled/filtered target position is then converted from ENU to ECEF coordinates:*

$$
\begin{bmatrix} X_a \\ Y_a \\ Z_a \end{bmatrix} = \begin{bmatrix} -\sin\lambda & -\sin\phi'\cos\lambda & \cos\phi'\cos\lambda \\ \cos\lambda & -\sin\phi'\sin\lambda & \cos\phi'\sin\lambda \\ 0 & \cos\phi' & \sin\phi' \end{bmatrix} \begin{bmatrix} x_a \\ y_a \\ z_a \end{bmatrix} + \begin{bmatrix} X_o \\ Y_o \\ Z_o \end{bmatrix} \tag{15}
$$

*where*

$\phi' = \tan^{-1}\left(\dfrac{Z_o}{\sqrt{X_o^2 + Y_o^2}}\right) =$ *geocentric latitude of ownship*

$(X_o, Y_o, Z_o) =$ *ECEF position of ownship*

$\lambda =$ *ownship longitude*

*Assembled/filtered target position in ECEF is then converted to WGS84 coordinates through a 15 step process:*

$$
\begin{aligned}
r_{ecef} &= \sqrt{X_a^2 + Y_a^2} \\
E^2 &= a^2 - b^2 \\
F &= 54 b^2 Z_a^2 \\
G &= r_{ecef}^2 + (1 - e^2) Z_a^2 - e^2 E^2 \\
C &= \frac{e^4 F r_{ecef}^2}{G^3} \\
S &= \sqrt[3]{1 + C + \sqrt{C^2 + 2C}} \\
P &= \frac{F}{3(S + \frac{1}{S} + 1)^2 G^2}
\end{aligned}
\tag{16}
$$

$$Q = \sqrt{1 + 2e^4 P}$$

$$r_0 = \frac{-(Pe^2 r_{ecef})}{1+Q} + \sqrt{\frac{a^2}{2}\left(1+\frac{1}{Q}\right) - \frac{P(1-e^2)Z_a^2}{Q(1+Q)} - \frac{\Pr_{ecef}^2}{2}}$$

$$U = \sqrt{(r_{ecef} - e^2 r_0)^2 + Z_a^2}$$

$$V = \sqrt{(r_{ecef} - e^2 r_0)^2 + (1-e^2)Z_a^2}$$

$$Z_0 = \frac{b^2 Z_a}{aV}$$

$$\phi = \tan^{-1}\left(\frac{Z_a + e'^2 Z_0}{r_{ecef}}\right)$$

$$\lambda = \tan^{-1}\left(\frac{Y_a}{X_a}\right)$$

$$h = U\left(1 - \frac{b^2}{aV}\right)$$

*where*

$a = semi\text{-}major\ axis\ =\ 6378137.0\ \ meters$

$b = semi\text{-}minor\ axis\ =\ 6356752.3142\ \ meters$

$e^2 = first\ eccentricity\ squared\ =\ 6.69437999014\ x\ 10^{-3}$

$e'^2 = second\ eccentricity\ squared\ =\ 6.73949674228\ x\ 10^{-3}$

$\phi = latitude$

$\lambda = longitude$

$h = altitude$

The track is then updated by modifying following parameters:

a. System time - updated with the TOA of the state vector report
b. Position – updated with assembled position in WGS84 coordinates
c. Velocity – updated with assembled velocity components
d. NIC – updated with NIC in the state vector report
e. State covariance matrices – updated with assembled covariance matrices

### A.2.2 Inter-Source Correlation

The Inter-Source Correlation function detects when an aircraft is tracked by multiple surveillance sources and prevents the display of ghost targets on the CDTI. Inter-Source Correlation consists of the following subfunctions:

a. Correlation between TCAS tracks and active ADS-B, ADS-R, and TIS-B tracks.
b. Correlation between active ADS-B/ADS-R and TIS-B tracks.
c. Correlation between an ownship track and TIS-B tracks.

#### A.2.2.1 TCAS to ADS-B/ADS-R/TIS-B Correlation

The TCAS to ADS-B/ADS-R/TIS-B correlation subfunction is event-driven by the reception of TCAS reports (or more precisely, initiated or updated TCAS tracks sent through the TCAS data bus). ASSAP attempts to correlate TCAS tracks with active ADS-B, ADS-R, and TIS-B tracks. Uncorrelated TCAS tracks are sent directly to the CDTI under the assumption that they represent the only surveillance source for the target. Figure A-4 contains a block diagram depicting the TCAS to ADS-B/ADS-R/TIS-B correlation function.

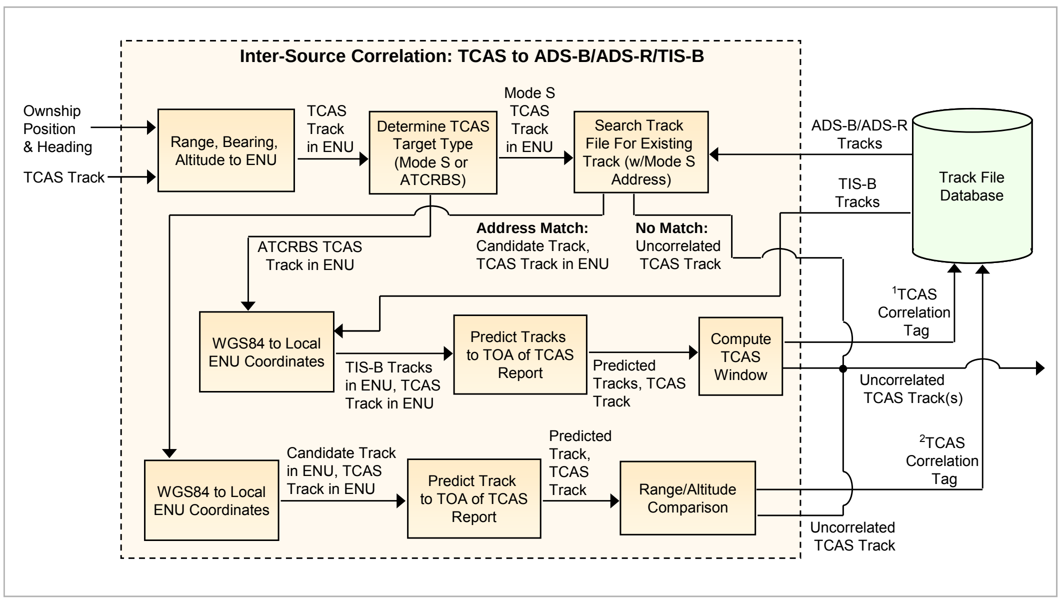


Figure A-4. Block diagram of the TCAS to ADS-B/ADS-R/TIS-B Correlation function.

*Notes:*

1) *For ATCRBS targets, a TCAS correlation tag is added to a track after repeated (i.e. 3) spatial correlations with a "complex" window; correlation history is stored in the track file.*
2) *For Mode S targets, a TCAS correlation tag is added to a track when its Mode S address matches that of a TCAS track and spatial correlation with a "simple" window succeeds.*

Upon reception of a TCAS report/track, ASSAP converts TCAS position (range, bearing, altitude) to local ENU coordinates:

$$
\begin{aligned}
z &= z_{tcas} - z_{own} \\
r_{xy} &= \sqrt{r^2 - z^2} \\
\alpha &= \theta - \phi \\
x &= r_{xy} \cos\alpha \\
y &= r_{xy} \sin\alpha
\end{aligned}
\tag{17}
$$

*where*

$z_{tcas}$ = *TCAS altitude*
$z_{own}$ = *ownship altitude*
$r$ = *slant (TCAS) range*
$r_{xy}$ = *horizontal range*
$\theta$ = *ownship heading (measured counter-clockwise from x)*
$\phi$ = *TCAS bearing (measured clockwise from ownship heading)*

Figure A-5 provides a pictorial depiction of the coordinate conversion.